Risk and Argument: A Risk-based Argumentation Method for Practical Security

Virginia N. L. Franqueira*, Thein Than Tun†, Yijun Yu†, Roel Wieringa* and Bashar Nuseibeh†‡
* University of Twente, Enschede, The Netherlands
Email: {franqueirav, r.j.wieringa}@ewi.utwente.nl
† The Open University, Milton Keynes, UK
Email: {t.t.tun, y.yu, b.nuseibeh}@open.ac.uk
‡ Lero, Irish Software Engineering Research Centre, Limerick, Ireland
Email: bashar.nuseibeh@lero.ie

***Abstract*—When showing that a software system meets certain security requirements, it is often necessary to work with formal and informal descriptions of the system behavior, vulnerabilities, and threats from potential attackers. In earlier work, Haley et al. [1] showed structured argumentation could deal with such mixed descriptions. However, incomplete and uncertain information, and limited resources force practitioners to settle for good-enough security. To deal with these conditions of practice, we extend the method of Haley et al. with risk assessment. The proposed method, RISA (RIsk assessment in Security Argumentation), uses public catalogs of security expertise to support the risk assessment, and to guide the security argumentation in identifying rebuttals and mitigations for security requirements satisfaction. We illustrate RISA with a realistic example of PIN Entry Device.**

***Index Terms*—Security Requirements, Argumentation, Risk Assessment, Common Attack Pattern Enumeration and Classification (CAPEC), Common Weakness Enumeration (CWE)**

I. Introduction

Structures of argumentation, such as those proposed by Toulmin et al. [2], provide an effective way to organize knowledge when convincing an audience about a claim. They have been used in many ways including to build safety cases [3], to demonstrate compliance to laws and regulations [4], and to show security requirements satisfaction [1]. Depending on the nature of the claim, arguments can be built upon observable and measurable evidence, such as the frequency of system failures, defects detected by code analysis tools and developers' certification records. Engineering of secure systems has many practical limitations, including incomplete knowledge about potential attackers, uncertainties about the system context, limited resources and different trade-off agendas. All in all, security is a non-zero-sum game between defenders and attackers of a system. Absolute security is usually not feasible, if at all possible. Argumentation works well in conditions of complete information and sufficient resources, but when one of these is lacking, it needs to be supplemented with risk assessment techniques that allow for uncertainty and incompleteness of available evidence, and allow partial application when resources are limited. It facilitates the achievement of *practical security*: "good-enough satisfaction" of security requirements within an "as low as reasonably practicable" [5] level of risk.

Haley et al. [1] have already suggested, but not elaborated, the need to relate argumentation and risk assessment for security. Extending that work, we present the RISA (RIsk-based Security Argumentation) method, which takes advantage of public catalogs of security expertise and empirical evidence, in particular, the CAPEC (Common Attack Pattern Enumeration and Classification), and the CWE (Common Weakness Enumeration) catalogs. The aims of RISA are to:

1) identify the risks to the satisfaction of security requirements;
2) differentiate between the risks that should be mitigated by the system context and the risks that should be mitigated by the system;
3) analyze those risks to be mitigated by the system and prioritize the risks found through arguments.

The main contribution of this paper is a systematic approach to assessing the risks associated with security arguments. This approach enables requirements engineers to make informed decisions about the implications of security risks, and how the system could maintain a good-enough level of security satisfaction. Our approach is different from other risk assessment frameworks in several ways. First, our approach uses security arguments, derived from a systematic description of the system context and formal reasoning about the satisfaction of the security requirements, as top-level structure of risk assessment. Second, RISA treats assumptions made in the arguments as a major source of risks, mitigation of which is a responsibility to be discharged either by the software or the system context. Third, the approach makes extensive use of publicly available catalogs on security risks, and mechanisms to mitigate risks. These evolving catalogs reflect the changing nature of security threats, and provide a valuable source of knowledge to requirements engineers.

The rest of the paper is organized as follows. Section II introduces our running example, the PIN Entry Device (PED) case study. Section III provides some background discussion on security requirements satisfaction, argumentation, and the security catalogs CAPEC and CWE that support the RISA method. Section IV presents the RISA method, describes the role played by argumentation and risk assessment, and lists its

steps. Section V demonstrates the RISA steps using the PED example. Section VI discusses the method and reviews related work, and Section VII concludes the paper.

II. Illustrative Case Study: PIN Entry Device

PIN Entry Device (PED) is a type of device widely-deployed and used by consumers to pay for goods with debit or credit smartcards at the Points-Of-Sales (POS).

When using the device, cardholders typically insert their cards, issued by a financial institution, into a card-reader interface of the PED, enter the PIN using the PED's keypad, and confirm the transaction value via a display on the PED itself. Then smartcard-based systems are expected to authenticate cardholders via the PIN and verify the card details against a public-key certificate before transactions can be completed successfully. These certificates are usually stored on the chip, but they can also be stored on the magnetic strip for compatibility with card-readers that have not adopted this technology. Most PEDs used in Europe implement the EMV (EuroPay, MasterCard and Visa) protocol in the process of authentication and authorization of payment transactions. This protocol drives the communication at the PED-card interface and the PED-bank interface. The protocol in principle allows only encrypted transmission of the PIN across these interfaces when the PED, card and bank support asymmetric cryptography. However, many card issuers adopt a low-cost EMV option in their smartcards that can be triggered to transmit unencrypted PIN on the interface PED-card.

Drimer et al. [6] studied two popular PEDs produced by different manufacturers, and evaluated either under the Visa evaluation scheme [7] or under the Common Criteria evaluation scheme [8]. The PEDs were found to be vulnerable to unsophisticated attacks in practice, and the authors have drawn lessons from this not only in relation to the evaluation process [6], [9] but also in relation to the software development life cycle. Throughout this paper, we will use this non-trivial example to explain and to motivate the RISA method.

III. Background

The RISA method relies on two main concepts proposed by Haley et al. [1]: the notion of the *satisfaction of security requirements*, and the use of *outer* and *inner* arguments to demonstrate the system security. This section recaps those concepts, and discusses the security catalogs used by the RISA method.

A. Satisfaction of Security Requirements

Following the WSR principle of the Problem Frames approach [10], the framework of Haley et al. separates software artifacts into W, S and R, where W represents a description of the world in which the software is to be used (i.e., the system context), S represents the specification of a system, and R represents a description of the requirements. The main property of these artifacts is that the software within the system context should satisfy the requirements, as indicated by the entailment (1):

$$W, S \vdash R \tag{1}$$

Similar to the Problem Frames approach, the framework of Haley et al. describes the world context W in terms of *domains* (short for *problem world domains*), elements of the world that can be either tangible (such as people, other systems, and hardware) or intangible (such as software and data structure). Typically, W also contains assumptions made about these domains. Domains interface with each other and with the system S via *shared phenomena* (such as events, states and values) that are controlled by a domain and are visible to some other domain(s).

In the framework of Haley et al., security requirements are constraints on functional requirements that protect the assets from identified harms. For example, in the requirement "Provide Human Resource (HR) data requested by the user, only to HR staff", providing HR data to users making the requests is regarded as a functional requirement, and ensuring that only those requests from members of HR staff are fulfilled is regarded as a constraint on the functional requirement, thus a security requirement. Security requirements are part of R in the entailment (1). Therefore, satisfaction of security requirements, spelled out in *security arguments*, show (i) whether properties of W and S entail the security requirements, and (ii) whether assumptions in W and S are correct.

B. Security Arguments of Haley et al. Framework

The framework of Haley et al. distinguishes two kinds of argument for satisfaction of security requirements.

1) Outer arguments: The *outer arguments* show whether properties of W and S entail the security requirements. These arguments are typically expressed in a formal language, such as propositional logic. Therefore, outer arguments are proofs of the entailment (1) for security requirements. This is expressed as follows:

$$(domain\ behavior\ premises) \vdash (security\ requirement(s)) \tag{2}$$

Outer arguments rely on properties of W and S (*domain behavior premises*), some of which may turn out to be incorrect assumptions. These premises need to be challenged, and be grounded in facts if possible, or taken as true, for instance, on the basis of a lack of contrary evidence.

2) Inner arguments: The general purpose of an *inner argument* is to try to rebut an outer argument by questioning its premises. Notice that the outer arguments establish the scope of security assessment, whilst the inner arguments deepen the assessment. The framework of Haley et al. uses Toulmin-style structures, enhanced with the notion of recursiveness from Newman et al. [11], for inner arguments. The general structure of inner arguments used in the framework of Haley et al. is shown in Fig. 1.

A **claim** is the object of an argument, a statement that one wishes to convince an audience to accept. A **ground** is a piece of evidence, a fact, a theory, a phenomenon considered to be true. A **warrant** is a statement that links a ground and a claim,

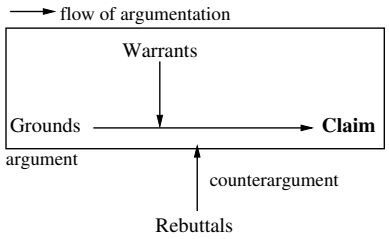

Fig. 1. Structure of arguments used in the framework of Haley et al. [1]

showing how a claim is justified by ground(s). A **rebuttal** is a counterargument which reduces support for the claim. Specifically in the case of security-related argumentation, rebuttals represent the *risks* that the claim of an argument is false. Rebuttals can be mitigated in order to restore the confidence that the claim of the rebutted argument is true. A mitigation, while negating a rebuttal, may introduce additional knowledge to show that the rebuttal, i.e. a risk, can somehow be tolerated. Therefore, mitigations address risks but may not necessarily eliminate the risks, and residual risks may remain. Moreover, mitigations may also introduce new risks, leading to new rounds of rebuttals and mitigations in argumentation.

C. Relationship between Outer and Inner Arguments

Fig. 2 shows how outer and inner arguments are related: the formal outer argument provides the main structure that drives the inner argumentation. Each of the outer argument premises $(a \rightarrow b)$ is the beginning for one thread of inner arguments, where a is the ground and b is the claim to be challenged by examining the ground a and the implication $\rightarrow$. Each round of inner argumentation is indicated by the notation $R \mid Mr.n$, where R stands for Risk and M for Mitigation, r indicates the round and n represents a sequential number.

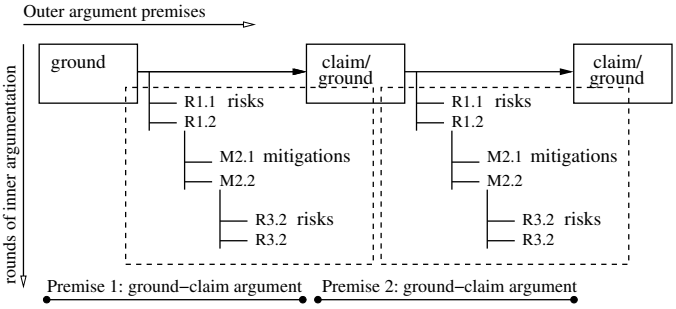

Fig. 2. Relationship between outer and inner arguments

In Section IV, we contrast the process of argumentation used in the framework of Haley et al. with our new approach proposed by the RISA method.

D. Public Catalogs of Security Expertise

The National Cyber Security Division of the U.S. Department of Homeland Security has a strategic initiative to promote software assurance. Two projects under this initiative are particularly useful for the RISA method, namely, CAPEC and CWE[1].

[1]Their websites, http://cwe.mitre.org/ and http://capec.mitre.org/, are both sponsored and managed by the Mitre Corporation.

Common Attack Pattern Enumeration and Classification (CAPEC) is a publicly available catalog of known attacks, described according to a standard schema [12]. The CAPEC catalog[2], contains 460 entries, organized according to different views and categories. Complete CAPEC entries provide not only information about attack execution flow, methods of attack, attack prerequisites, and attacker skills/knowledge/resources required, but also about typical severity, and generic solutions and mitigations. In addition, complete entries in CAPEC also provide pointers to specific weaknesses ("underlying issues that may cause vulnerabilities" [12]), i.e. to CWE(s), and to concrete examples of vulnerabilities which have been detected in use (these are CVE (Common Vulnerabilities and Exposure) entries recorded in the National Vulnerability Database[3]).

Common Weakness Enumeration (CWE) is a public catalog of weaknesses[4], where each weakness can either become a direct source of vulnerabilities, or contribute indirectly to an increase in the likelihood of an attack to happen and/or an increase in the impact of the attack, if it succeeds [12]. It also follows a standard schema [13], providing different views, groupings and relations with other weaknesses, as well as pointers to CAPEC and CVE entries. Complete CWEs indicate common consequences, likelihood of exploit, detection methods and potential mitigations at different phases, such as Requirements and Architecture/Design. The RISA method uses CAPEC and CWE as sources for security knowledge, making the analysis of risks more systematic, repeatable, and less dependent on subjective judgments.

IV. Overview of the RISA Method

As shown in Fig. 3 the RISA (RIsk assessment in Security Argumentation) method extends the process of argumentation for security requirements proposed in the Haley et al. framework by incorporating a process of risk assessment.

Steps 1 to 3 of the proposed approach are same as the first three steps of the framework of Haley et al., and are briefly summarized below.

A. Step 1 to Step 3

In Step 1 (Identify Functional Requirements), functional requirements of the system and the system context (domains and shared phenomena) are identified. These requirements may be derived from the higher-level goals of the system. In Step 2 (Identify Security Goals), assets that need to be protected, management principles regarding those assets, and security goals are identified. In Step 3 (Identify Security Requirements), security requirements are derived from security goals, and are expressed as constraints on the functional requirements identified in Step 1. Problem diagrams are constructed in this step.

[2]http://capec.mitre.org/data/, version 1.6, accessed on 21 Feb 2011
[3]http://nvd.nist.gov/, version 2.0, accessed on 21 Feb 2011
[4]http://cwe.mitre.org/data/, version 1.11, accessed on 21 Feb 2011

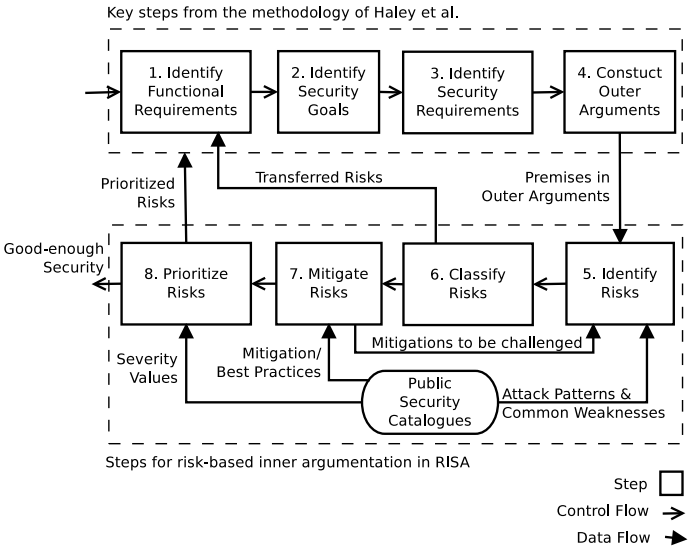

Fig. 3. Schematic overview of the RISA method

B. Step 4: Construct Outer Arguments

Unlike the fourth step of the Haley et al. framework, only the outer arguments for security requirements (excluding the inner arguments) are constructed in Step 4 of RISA. These outer arguments are formal, and they make use of domain properties, correctness of which is examined by inner arguments. Behavioral premises used in the outer arguments may represent risks, which are identified using a systematic risk assessment process in RISA. This is represented in the figure by the arrow from Step 4 to Step 5.

Steps 5 to 8 correspond with the process of constructing inner arguments in the Haley et al. framework. These four steps show how domain assumptions in outer arguments are challenged by means of risk assessment based on public security catalogs.

C. Step 5: Identify Risks

In this step, behavioral premises in outer arguments regarding the domains (arrow from Step 4 to Step 5 in Fig. 3) are identified as potential risks. For instance, in the PED example, there could be a behavioral premise about the confidentiality of the PIN entered using the keypad. Public security catalogs are then searched to find known security weaknesses regarding the confidentiality of passwords entered using a keypad.

D. Step 6: Classify Risks

The risks are classified into two groups. In one group are risks transferred to the context because nothing can be done by the system to mitigate them. In the PED example, the confidentiality of the PIN (one of the PED security requirements) depends on cards having certain cryptographic capabilities. Although the PED can generally comply with this demand, there is still the risk that the card will not comply. Therefore, the obligation to mitigate this risk is transferred either partially or fully to the card. In the other group are risks that should be mitigated by the system. In the PED example, integrity of the PIN over the network has to be ensured by the system. Classification and transferring of obligations to mitigate risks to the system and its context could modify the behavior and properties of the domains and shared phenomena, and perhaps even the functional requirements (as indicated by the arrow from Step 6 to Step 1). As a result of these changes, outer arguments may be rebutted.

E. Step 7: Mitigate Risks

Appropriate security mechanisms for mitigating the risks are searched for in the public security catalogs (arrow from the catalogs to Step 7). Some of these mechanisms themselves could introduce new risks and therefore should be assessed in a new round of inner argumentation (arrow from Step 7 to Step 5).

F. Step 8: Prioritize Risks

In the last step, risks are prioritized on the basis of their severity as indicated by the public security catalogs (arrow from catalogs to Step 8). These risks affect the priority of requirements to be satisfied (arrow from Step 8 to Steps 1–4). When the residual risks are deemed to be acceptable given the limitation of development resources, the system has reached the level of good-enough security.

G. Discussion

Fig. 4 illustrates how elements from risk assessment, used in the RISA method, fit into the inner argument schema, as presented in Fig. 2. Unlike the traditional process of argumentation (based on Toulmin's practical argumentation [2]), that is intrinsically performed in depth-first as a dialog (Fig. 2), argumentation based on risk assessment is performed in breadth-first. Each of these approaches have advantages (discussed in Section VI) but, typically, in risk assessment two rounds of argumentation are performed within the same cycle: one is related to risks that rebut outer arguments, and the other is related to mitigations that counter these risks.

Therefore, risks regarding all behavioral premises are identified and classified in terms of those that should be mitigated by the system and those that should be mitigated by the context, and prioritized on the basis of the severity of the risk they represent, providing input to the next round of argumentation.

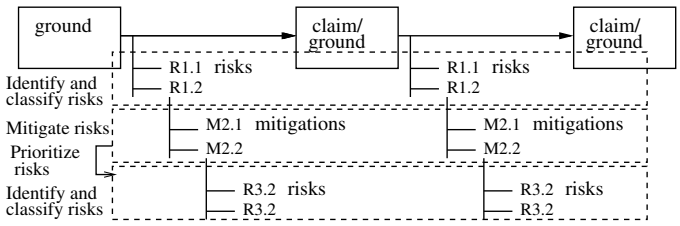

Fig. 4. Risk-based inner argumentation

The recursiveness of the inner argumentation is represented by the indirect connection between Step 5 and Step 7, which involves the process of finding mitigations to risks, and the

direct connection between Step 7 and Step 5, which involves the process of finding new risks in mitigations.

Public catalogs provide input for all steps in risk assessment, except for Step 6, which has to be done by domain experts relying on the knowledge of the exact requirements. In the RISA method, we use CAPEC and CWE to feed the identification of risks with descriptions and information about known attack patterns and weaknesses in software. They can also provide information on how these attacks and weaknesses can be mitigated, and empirical values indicating the severity that allow the prioritization of risks. In some cases, in-house security catalogs may supplement public security catalogs.

Since these risks are attached to arguments and security requirements, prioritizing risks indirectly results in the prioritization of arguments and security requirements.

V. The PED Example

This section describes a step-by-step application of RISA to the PED example.

A. Satisfaction of Security Requirements

The system under analysis is the PED; it consists of four main components: card-reader, keypad, display and CPU.

1) Step 1—Identify Functional Requirements: The overall functional goal of the system in relation to PED users is the following:

[FG1]: Provide convenient payment option at Points-Of-Sale to consumers

The following functional requirement can be derived from the functional goal above:

[FR1]: Allow consumers to pay at Points-Of-Sale with PIN

2) Step 2—Identify Security Goals: Obviously, the consumers' PIN is the most valuable asset in this case. Protecting the PIN is important not only because of the potential financial impact on the consumers, but also because of the substantial negative impact it will have on the reputation of the banking authorities [14], and the PED manufacturers. Card details, stored in smartcards, are also relevant assets; their value comes from the possibility to fake the magnetic-strip on the smartcards [6]. Other assets involved with the PED system include: transaction value, design characteristics, the smartcard itself, and cryptographic keys. For the illustration of our analysis, the main security goal is to protect the PIN.

3) Step 3—Identify Security Requirements: As mentioned earlier, security requirements are viewed as constraints on the system functional requirement according to its functional goals. In this case, such a composition produces the following two security requirements:

[SR1]: PIN entered by consumers shall remain confidential during payment transactions at Points-Of-Sale [SR2]: PIN entered by consumers shall remain accurate during payment transactions at Points-Of-Sale (i.e. integrity of the PIN shall be preserved)

According to the documentation available about the PED system (e.g. [6], [14]), these two requirements are satisfied by implementing the following security functions:

[SF1]: Enclosure of PED components provides tamper detection and response mechanisms to resist physical attacks [SF2]: Encryption/Decryption of PIN ensures that the PIN is encrypted within the PED immediately after the PIN entry is complete

These security functions correspond with the "requirements" A1.1 and C2, extracted from the PED security requirements document from Mastercard [15].

The system context, W in the entailment (1), is then elaborated. The functional requirement of the PED helps us to delimit the context; from "payment transaction using a PED" we identify five domains: consumer, card, merchant, terminal and bank. Using a slightly modified version of the notation used by the Problem Frames approach [10], Fig. 5 shows the context of the PED system and its security requirements. The notation is unusual in two ways: (i) it treats the PED system as a machine with its own components, and (ii) causality in shared phenomena is indicated by directed arrows. Notice that the diagram shows the shared phenomena related not only to PIN, but also to the card details and the transaction value, which are relevant to the PED payment transactions.

In terms of PIN, the diagram illustrates the behavior already described in Section II, namely that "Cardholders [consumers] typically insert their cards ... into the PED's card-reader interface, enter their PIN using the PED's keypad, and confirm the transaction value via a display on the PED itself". Note that the action performed by consumers to insert their cards into the card-reader is not represented explicitly in the diagram.

The PED system has two possible main usage scenarios. It can be used at the Points-Of-Sales connected to a terminal, or stand-alone. The former case is often found in supermarkets, once the merchant scans products, their value get registered by the terminal; at the end, the terminal sends the value of the transaction to the PED that displays the value to the consumer. The latter case is often found in restaurants: the merchant enters the transaction value directly into the PED via the keypad and the PED displays this value to the consumer.

The card details flow from their initial location (the card) along with the PIN and the transaction value until they reach the bank for approval.

The shared phenomena related to the PIN in Fig. 5 are detailed using the following convention: an arrow from a domain A annotated with phenomenon b is written as $A!b$.

- consumer!PIN: consumer enters PIN
- keypad!PIN: keypad sends PIN to the card-reader
- card-reader!PIN: card-readers sends PIN to the card
- card!confirmation-PIN-ok: card requests confirmation that PIN is ok to the card-reader
- card-reader!PIN-confirmed(PIN,card-details): card-reader sends confirmation that PIN is ok to the PED CPU
- CPU!authorization-request(PIN,value,card-details): CPU sends request for authorization of payment transaction to bank
- bank!confirmation-transaction: bank sends confirmation (positive or negative) of transaction to the PED CPU

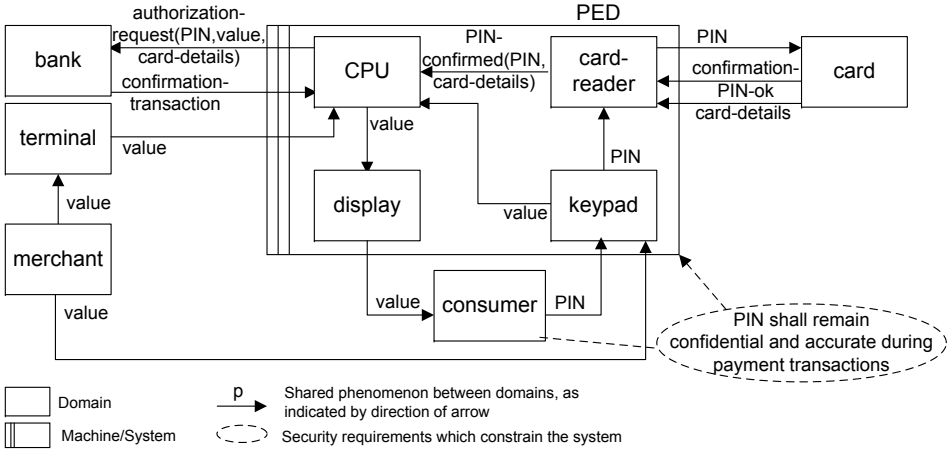

Fig. 5. System context of the PED system and its security requirements

4) Step 4—Construct Outer Arguments: The behavior of the PIN needs to guarantee that:

P1, P2, P3, P4, P5, P6, A7 ⊢ bank!confirmation-transaction

The premises are defined below using the propositional logic.

Premises:
P1. consumer!PIN → keypad!PIN
P2. keypad!PIN → card-reader!PIN
P3. card-reader!PIN → card!confirmation-PIN-ok
P4. card!confirmation-PIN-ok → card-reader!PIN-confirmed
P5. card-reader!PIN-confirmed → CPU!authorization-request
P6. CPU!authorization-request → bank!confirmation-transaction
Triggering assumption:
A7. consumer!PIN holds
Conclusions:
C8. keypad!PIN (Detach,P1,A7)
C9. card-reader!PIN (Detach,P2,C8)
C10. card!confirmation-PIN-ok (Detach,P3,C9)
C11. card-reader!PIN-confirmed (Detach,P4,C10)
C12. CPU!authorization-request (Detach,P5,C11)
C13. bank!confirmation-transaction (Detach,P6,C12)

Assuming that the phenomena in behavioral premises (P1–P6) are triggered in sequence, context is correct and later implementation of the PED does not introduce deviations from the behavior specified, this proof means that, *in principle*, the PED behavior (i.e., the PED with its security functions under its context) can satisfy both security requirements (SR1: confidentiality of PIN, and SR2: accuracy of PIN) while meeting its own functional requirement (FR1). It proves that the entailment (2) can be fulfilled for the PED example.

However, the proof premises (P1-P6) and triggering assumption (A7) can be challenged in practice. If any of these can be challenged effectively, the result could represent a non-satisfaction of entailment (2) for the PED example.

B. Risk assessment for inner arguments

The premises (P1–P6) and triggering assumption (A7) of the outer argument, the output of the previous step, provides a structure for assessing risks. The following discussion will focus on risks related to SR1 only.

1) Step 5—Identify Risks: This activity aims to identify the risks related to the security requirement SR1 that could rebut all the premises and triggering assumptions carried over from the outer argument. Supported by the CAPEC and CWE security catalogs, the activity involves searching for catalog entries that represent a risk to the claim of each premise, including the triggering assumption. For example, what has to be challenged for premise "P1: consumer!PIN → keypad!PIN" is the confidentiality of consumers' PIN (SR1). This involves: (i) the confidentiality of the PIN as it is entered by consumers (A7), and (ii) the confidentiality of the PIN from the moment it is entered by consumers until it reaches the keypad. Therefore, these two aspects drive the analysis of risks which can rebut P1. As illustrated in Fig. 6, risks identified for P1, given A7, allow us to evaluate the satisfaction of conclusion C8 of the outer argument for SR1. Similar rationale applies to P2-P6.

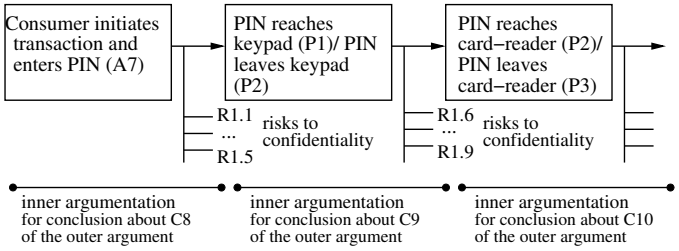

Fig. 6. Behavioral premises of the PED system to be challenged via risk-based inner argumentation

Table I lists the risks identified for premises P1 and P2 with references to CAPEC and CWE entries. It also contains some references marked with the symbol ⋆, indicating the CAPEC/CWE entries that are related to the risk but are not incomplete enough to be useful at this stage. Since these catalogs are constantly evolving it is important to keep them for future reference.

2) Step 6—Classify Risks: We classify risks according to two types of risk treatments.

TABLE I
IDENTIFICATION OF RISKS FOR PREMISES P1 AND P2

Challenged	Risk	Reference
Premise P1	R1.1: consumer is triggered to reveal PIN via social engineering attack	CAPEC-403⋆
Premise P1	R1.2: PIN is revealed by missing PIN field masking	CWE-549
Premise P1	R1.3: PIN is revealed by brute force attack	CAPEC-49, CAPEC-70 & CAPEC-112
Premise P1	R1.4: PIN is revealed due to lack of aging policy	CWE-262
Premise P1	R1.5: PIN is collected by fake PED set to allow pharming attack	CAPEC-89
Premise P2	R1.6: PIN is revealed if sent unencrypted within the PED and the PED enclosure can be tampered with	CWE-311 & CAPEC-436⋆
Premise P2	R1.7: PIN is revealed if sent encrypted within the PED but PED enclosure can be tampered with	CAPEC-20 & CWE-327 & CAPEC-436⋆
Premise P2	R1.8: PIN is revealed via sniffer installed by PED administrators	CAPEC-65
Premise P2	R1.9: Unauthorized access to PIN is concealed via log injection-tampering-forging by PED administrators	CAPEC-93

A risk is classified as "transfer risk" if it is assumed that context domains, involved in ensuring the satisfaction of the entailment (1), will be responsible for its mitigation. Identifying such a class of risks is important for two reasons: it allows different parties in the PED context domains to be made explicitly accountable for the mitigation of the identified risks, and it allows regulations to be enforced [16].

A risk is classified as "mitigate risk" if it is assumed that the system will be responsible for its mitigation. The classification of risks from Table I is shown in Table II. Note that this table illustrates with R1.7 that one risk can be classified in both classes. Typically, risks classified under "mitigate risk" contain brief descriptions of mitigations in the format of requirements. Mitigations described in this table are mostly retrieved from best practices *solutions and mitigations* described in the CAPEC and CWE entries mentioned in Table I. For example, CWE-311 (reference for risk R1.6), entitled "Missing Encryption of Sensitive Data" discusses potential mitigations for this weakness in terms of different phases of the software life cycle. Under *Phase:Requirements*, we find the description of mitigation "Any transmission of PIN should use well-vetted encryption algorithms", as presented in Table II.

3) Step 7—Mitigate Risks: Only those risks classified as "mitigate risks" are carried over to this stage. This step involves a cross-analysis of mitigations identified for each risk during the classification activity (Table II), in order to obtain

TABLE II
CLASSIFICATION OF RISKS FOR PREMISES P1 AND P2

Risk	Risk treatment
R1.1	**Transfer risk**: assumed consumers take mitigations
R1.2	**Mitigate risk**: PED should obfuscate display of PIN as entered by consumers in keypad
R1.3	**Transfer risk**: assumed banks (e.g., require strong PIN policy) and consumers (e.g., avoid common, guessable PIN) take mitigations
R1.4	**Transfer risk**: assumed banks and card issuers take mitigations (e.g., by periodically requiring PIN change and card renewal)
R1.5	**Mitigate risk**: PED should use (i) authentication mechanisms, and (ii) audit mechanisms to log authorized replacements
R1.6	**Mitigate risk**: Any transmission of PIN should use well-vetted encryption algorithms
R1.7	**Mitigate risk**: (i) encryption of PIN should use accepted algorithms and recommended key sizes, (ii) cryptographic keys should be managed and protected (**Transfer risk**: assumed cards will also comply with this mitigation), (iii) PED design should allow upgrade of cryptographic algorithms
R1.8	**Mitigate risk**: Any transmission of PIN should be encrypted
R1.9	**Mitigate risk**: PED should provide access control to physical log files

a consolidated list of mitigations, as shown in Table III. This table shows that mitigation M2.4 counters risks R1.6, R1.7 and R1.8, and that risk R1.7 is countered by mitigations M2.4, M2.5 and M2.6. Notice that risks R1.1., R1.3 and R1.4 are not presented in the table.

TABLE III
MITIGATIONS OF RISKS IDENTIFIED FOR P1 AND P2

Risk	Mitigation
R1.2	M2.1: PED should obfuscate display of PIN as entered by consumers in keypad
R1.5	M2.2: PED should use authentication mechanisms
R1.5	M2.3: PED should have audit mechanisms to log authorized replacements
R1.6 & R1.7 & R1.8	M2.4: Any transmission of PIN should use well-vetted encryption algorithms and recommended key sizes
R1.7	M2.5: Cryptographic keys should be managed and protected
R1.7	M2.6: PED design should allow upgrade of cryptographic algorithms
R1.9	M2.7: PED should provide access control to physical log files

4) Step 8—Prioritize Risks: This step involves retrieving from the CAPEC and CWE entries indicating the severity of risks identified, as shown in Table IV.

This table makes it evident that this analysis only gives a rough impression of the severity of risks and that security experts are again confronted with incomplete information, which often happens in security practice. It also becomes apparent that, because several CAPEC and CWE entries may

TABLE IV
PRIORITIZATION OF RISKS FOR PREMISES P1 AND P2

Mitigation	Risk	Typical risk severity
M2.1	R1.2	no indication in the CWE
M2.2	R1.5	very high
M2.3	R1.5	
M2.4	R1.6 & R1.7 & R1.8	low to very high (depending on specifics of different attacks)
M2.5	R1.7	low to high (depending on specifics of different attacks)
M2.6	R1.7	
M2.7	R1.9	high

refer to the same risk, we may obtain in the end a lower bound and an upper bound on risk severity. In this case, a decision should be made about the strategy to follow depending on several factors, such as attitude towards risk (e.g., risk-averse, risk-taking or in-between), security-criticality of the system, and so on. For example, from a risk-averse point-of-view, we consider M2.2, M2.3 and M2.4 as priority over the others.

Recursion of Inner Arguments Mitigations may also introduce new risks, so for each mitigation in Table III a new iteration of Step 5 may be required. These iterations stop when the system security is considered good-enough, i.e. the residual risks are considered acceptable, and the resources for security analysis have been used (e.g., deadline or budget have been reached).

For example, if we take mitigation M2.3 which is related to the very high severity risk R1.5 (Table IV), we can see that this risk refers to the possibility of pharming attacks (Table I). According to CAPEC-89, a pharming attack occurs "when the victim is fooled into entering sensitive data into supposedly trusted locations". This risk challenges the premise P1, which refers to the PIN shared between consumer and keypad (Fig. 5). Since SR1 is about the confidentiality of PIN, the risk assessment suggests that the pharming attack is a rebuttal to the security argument of SR1. The risk assessment further indicates that one way to mitigate this risk is by introducing an audit mechanism for the keypad (Table III). This new round of rebuttal to the premises of the security argument, and a mitigation to the rebuttal are obtained from the risk assessment.

VI. RELATED WORK & DISCUSSION

A. Risk in Requirements Engineering

In requirements engineering literature, the issue of risks has been considered in two related yet distinct ways: project risks and system risks. Project risks are related to the process of software development and factors that may contribute to the failure and success of the project. Several factors may be related to the project risks, including requirements creep [17], [18], requirements negotiation and project estimates [18], and project resources. System risks are related to the behavior of the software system that may contribute to the system satisfying or not satisfying the requirements. Factors contributing to system risks include missing or incorrect requirements and domain assumptions [19].

Threat modeling for the elicitation of security requirements has been extensively researched in the domain of requirements engineering. Published approaches include: misuse cases [20], abuse cases [21], attack trees [22], abuse frames [23], anti-goals [24], and combinations of these [25]. These approaches overlap only partially with the RISA method, mainly in the activity of risk identification. In this activity, they may provide a more in-depth analysis of specific issues/scenarios, therefore, in this sense they complement RISA. However, they lack the advantage of argumentation to allow validation of security requirements satisfaction and of maintaining the focus of security on the system as a whole.

B. Structured Argumentation

Argumentation provides a rationale to convince an audience that a claim should be considered valid. Three qualities are often discussed in the informal argumentation literature: convincingness, soundness, and completeness. Convincingness relates to whether the argumentation is compelling enough to assure an intended audience that the conclusion reached is reasonable [1]. Soundness relates to whether the argumentation fulfills the argumentation schema and is based on "true premises" [26]. Completeness relates to whether nothing has been omitted that could lead to a different conclusion about a claim [26], [27].

A known problem in argumentation is the subjectivity involved in identifying arguments and counterarguments (which relates to soundness), and the difficulty in determining completeness. Proposals to reduce these problems rely on the help of: (i) pre-defined critical questions [28], [29], (ii) what-if scenarios [30], (ii) expert assurance checks [26], (v) guidelines [31] or (vi) how/why questioning, as proposed in [1]. However, these approaches provide limited support and are rather static, i.e. they do not evolve in the speed required to assure good-enough security of systems. The RISA method reduces these problems by using ever evolving public catalogs, updated using input by a pool of security experts from several organizations[5].

Since security involves uncertainty and incomplete information, it becomes difficult, if not impossible, to show satisfaction of these qualities. Expert judgment is needed, probably enhanced with peer review [26], to improve the quality of *good-enough* security argumentation. Nevertheless, the main benefit of using argumentation is that, in contrast to an ad-hoc approach, it structures the reasoning and exposes it to criticism [26]. The RISA method contributes towards reducing incompleteness and uncertainty because its risk-based argumentation is supported by public catalogs which accumulate information about risks, best practice mitigations and empirical severity values based on input from a large community of security experts.

[5]http://cwe.mitre.org/community/index.html, accessed on 21 Feb 2011

C. Traditional versus risk-based (security) argumentation

As mentioned in Section III, the process of argumentation traditionally follows a depth-first style. It provides a neat and intuitive view about the evolution of an argument in the format of a debate between two opponents. It is well-suited to be represented as a tree structure of arguments and counterarguments.

This process of argumentation, however, is not suitable when reasoning about practical security. For example, it often happens that one mitigation counters several risks, one risk challenges several premises, and several mitigations introduce one same new risk as seen in Table III, Step 2. As a result, rather than a tree, a more complex graph structure of arguments is often needed. In such situations, a breadth-first style of argumentation based on risk assessment scales better. The RISA method is designed for such practical needs, while still providing the ability to reconstruct argument threads via backward traceability. Thus it is possible to link mitigations back to risks, then back to premises, which relate to the outer arguments to be validated in the first place.

D. Risk Assessment

Three basic elements distinguish the risk-based security argumentation method described in this paper from other risk assessment frameworks: (i) it provides a rationale for a systematic representation of system context from which the behavior of the system is derived and the outer argument is constructed, providing structure to the risk assessment part of the method, (ii) it makes assumptions about domains of the system context explicit via transferred risks, and (iii) it uses public security catalogs to support the risk assessment.

Explicit Context. In the CORAS framework [32], context is established by means of an asset diagram (CORAS Step 2) which contains all assets that are logically or physically related to the system to be assessed. Assets that are directly harmed as a consequence of an unwanted event affecting the target of evaluation are considered as part of the system under analysis: otherwise, they are considered as part of the system context. The boundary between system and system context can be further adjusted in CORAS (Step 3) after a preliminary analysis of risks by means of "dependent diagrams" [33]. Context is delimited in the RISA method by means of the system functional goal and security requirements, from which context domains are derived. This allows us to relate the results of risk assessment to the satisfaction of security requirements.

Other security risk assessment and evaluation frameworks (e.g., CRAMM [34], ISO 27005:2008 [35], AS/NZS4360:2004 [36], and Common Criteria [8]) do not prescribe any representation or delimitation rationale for context specification; context is often described in natural language.

Assumptions as Risk. Among the above mentioned risk assessment and evaluation frameworks, the Australian/New Zealand Standard (AS/NZS4360:2004 [36]) is the only one which conveys explicitly the idea that assumptions represent risks. It prescribes that assumptions should be recorded and clearly stated but, most importantly, mandates sensitivity analysis to test the effect of uncertainty in assumptions. The current RISA method does not incorporate validation of assumptions made about the system context but it classifies them as risks to be transferred. We plan to further study risks transferred and how they affect the satisfaction of security requirements as future work.

Evolving & Detailed Source of Information. The CORAS framework [32] uses the guidelines in ISO 27001:2008 [35] to support risk assessment. CRAMM [34] is supported by a proprietary database of security controls that are traceable to risks derived also from the ISO 27001:2008 [35]. Although ISO 27001 is publicly available, it is a static document and does not provide the level of details provided by CAPEC and CWE catalogs. However, both CAPEC and CWE, at their current stage, still need a lot of improvements, especially in their search capabilities. This is an area that has started to receive attention from researchers, see e.g. [37].

Specific RISA steps, such as "Identify Risks" and "Mitigate Risks", could be supported by other risk assessment frameworks such as the model-based approach of CORAS which uses icons instead of tables. However, although it may improve the user-friendliness of RISA, the traceability of several rounds of argumentation may become more complex.

VII. Conclusion and Future Work

Although absolute security is not possible in practice, security requirements still have to be satisfied to the extent allowed by incomplete information, uncertainty and limited resources. When dealing with practical security, requirements engineers need to reason about whether security is good enough. That reasoning typically involves risk assessment. Extending existing work on argumentation for security, the proposed RISA method has shown how argumentation can be extended with risk assessment, thereby exploiting the ability to identify, classify, mitigate and prioritize risks, and feeding the prioritized risks back to the process of reasoning about the satisfaction of the security requirements. RISA takes advantage of publicly available catalogs of common attacks and weaknesses, thus a degree of objectivity can be achieved in the risk assessment. Nevertheless, subjectivity is not completely eliminated by catalogs such as CWE and CAPEC: for instance, the evaluation of risk severity is likely to remain subjective. However, this evaluation is prone to scrutiny by security experts from the wider community.

The most pressing issue in our future work is validation. We have demonstrated an application of RISA to a realistic example of the PIN Entry Device. We are planning to apply the RISA method to significant industrial case studies, including the Air Traffic Management system [38]. Furthermore, we see two main directions for future work which complement each other: enhancements to the prioritization of risks and mitigations, consequently reflecting on the prioritization of arguments, and tool support.

In the current version of the RISA method, prioritization of risks is qualitative, coarse-grained, and is detached from

context. We believe that the Common Weakness Scoring System [39], which is another recent initiative related to the CWE, may be valuable in working towards a more systematic quantitative prioritization of risks taking into account factors related, for example, to security concerns particular to a business domain and technology. Prioritization could also be enhanced by incorporating cost/benefit trade-off considerations (as in [40]) about mitigations, as well as consideration about risks. Finally, uncertainty is another important aspect to be added into prioritization: early work on Probabilistic Argumentation [41] seems to be a promising starting point in this direction.

Decision making tool support would greatly improve the practical use of the RISA method. We plan to investigate how to provide effective automated reasoning about security requirements, based on feedback from risk-based argumentation.

ACKNOWLEDGMENT

The first author is supported by the research program Sentinels (http://www.sentinels.nl). The UK-based authors are supported by the SecureChange project, and SFI grant 03/CE2/I303_1. We thank the anonymous reviewers for helpful comments and suggestions.

REFERENCES

[1] C. Haley, R. Laney, J. Moffett, and B. Nuseibeh, "Security Requirements Engineering: A Framework for Representation and Analysis," *IEEE Trans. Softw. Eng.*, vol. 34, no. 1, pp. 133–153, 2008.

[2] S. Toulmin, R. Rieke, and A. Janik, *An Introduction to Reasoning*. Macmillan, 1979.

[3] T. P. Kelly, "Arguing Safety - A Systematic Approach to Safety Case Management," Ph.D. dissertation, University of York, 1998.

[4] B. Burgemeestre, J. Hulstijn, and Y.-H. Tan, "Value-Based Argumentation for Justifying Compliance," in *DEON'2010*. Springer, 2010, pp. 214–228.

[5] R. A. Weaver, "The Safety of Software: Constructing and Assuring Arguments," Ph.D. dissertation, University of York, September 2003.

[6] S. Drimer, S. J. Murdoch, and R. Anderson, "Thinking Inside the Box: System-Level Failures of Tamper Proofing ," in *SP'2008*. IEEE Press, May 2008, pp. 281–295.

[7] Visa, "Testing and Approval," Website, 2011, https://partnernetwork.visa.com/vpn/global/category.do?categoryId=103&documentId=501&userRegion=1, last visited Feb 2011.

[8] "Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision 3, CCMB-2007-09-001, CCMB-2007-09-002 and CCMB-2007-09-003," July 2009.

[9] S. Drimer, S. J. Murdoch, and R. Anderson, "Failures of Tamper-Proofing in PIN Entry Devices," *IEEE Security and Privacy*, vol. 7, pp. 39–45, November 2009.

[10] M. Jackson, *Problem Frames: Analysing and Structuring Software Development Problems*. Addison-Wesley/ACM Press, 2001.

[11] S. E. Newman and C. C. Marshall, "Pushing Toulmin Too Far: Learning from an Argument Representation Scheme," Xerox PARC, Tech. Rep. SSL-92-45, 1991.

[12] S. Barnum, *Common Attack Pattern Enumeration and Classification (CAPEC) Schema Description*, Copyright Cigital Inc., commissioned by the U.S. Department of Homeland Security, January 2008, http://capec.mitre.org/documents/documentation/CAPEC_Schema_Description_v1.3.pdf, Version 1.3.

[13] CWE Team, "CWE Schema Documentation," Online by The MITRE Corporation, December 2010, http://cwe.mitre.org/documents/schema/index.html, Version 4.4.2.

[14] The-Card-Payment-Group, "PIN Entry Device Protection Profile," Common Criteria Portal, Jul 2003, www.commoncriteriaportal.org/files/ppfiles/PED_PPv1_37.pdf, last visited Jul 2010.

[15] Mastercard International Incorporated, "Payment Card Industry POS PIN Entry Device Security Requirements," m2m Group website, http://www.m2mgroup.ma/livresetdocs/security%20risk.htm, last visited Feb 2011, October 2004, version 7 1.0, Revised March 2005.

[16] R. Anderson, "Failures on Fraud," *Speed*, vol. 3, no. 2, pp. 6–7, September 2008.

[17] R. A. Carter, A. I. Antón, L. A. Williams, and A. Dagnino, "Evolving Beyond Requirements Creep: A Risk-Based Evolutionary Prototyping Model," in *RE'01*, 2001, pp. 94–101.

[18] J. Chisan and D. Damian, "Exploring the role of requirements engineering in improving risk management," in *RE'05*, 2005, pp. 481–482.

[19] A. V. Miranskyy, N. H. Madhavji, M. Davison, and M. Reesor, "Modelling Assumptions and Requirements in the Context of Project Risk," in *RE'05*, 2005, pp. 471–472.

[20] G. Sindre and A. L. Opdahl, "Eliciting security requirements with misuse cases," *Requirements Engineering Journal*, vol. 10, no. 1, pp. 34–44, 2005.

[21] J. McDermott and C. Fox, "Using Abuse Case Models for Security Requirements Analysis," in *ACSAC'99*. IEEE Press, 1999, pp. 55–64.

[22] B. Schneier, "Attack Trees: Modeling Security Threats," *Dr. Dobb's Journal*, December 1999.

[23] L. Lin, B. Nuseibeh, D. Ince, and M. Jackson, "Using abuse frames to bound the scope of security problems," in *RE'04*. IEEE Computer Society, 2004, pp. 354–355.

[24] A. van Lamsweerde, "Elaborating Security Requirements by Construction of Intentional Anti-Models," in *ICSE '04*. IEEE Press, 2004, pp. 148–157.

[25] I. A. Tøndel, J. Jensen, and L. Røstad, "Combining Misuse Cases with Attack Trees and Security Activity Models," in *ARES'2010*. IEEE Press, 2010, pp. 438–445.

[26] P. Graydon and J. Knight, "Success Arguments: Establishing Confidence in Software Development," University of Virginia, Tech. Rep. CS-2008-10, July 2008.

[27] S. B. Shum and N. Hammond, "Argumentation-based Design Rationale: What Use at What Cost?" *Int. Journal of Human-Computer Studies*, vol. 40, no. 4, pp. 603–652, 1994.

[28] D. N. Walton, *Argumentation Schemes for Presumptive Reasoning*. Mahwah NJ, USA: Lawrence Erlbaum Associates, 1996.

[29] K. Atkinson, T. Bench-Capon, and P. McBurney, "Justifying Practical Reasoning," in *CMNA'04*, 2004, pp. 87–90.

[30] P. Baroni, F. Cerutti, M. Giacomin, and G. Guida, "An Argumentation-Based Approach to Modeling Decision Support Contexts with What-If Capabilities," in *AAAI Fall Symposium. Technical Report SS-09-06*. AAAI Press, 2009, pp. 2–7.

[31] H. Lipson and C. Weinstock, "Evidence of Assurance: Laying the Foundation for a Credible Security Case," May 2008, department of Homeland Security; online: https://buildsecurityin.us-cert.gov/bsi/articles/knowledge/assurance/973-BSI.html, last visited Feb 2011.

[32] F. den Braber, I. Hogganvik, M. S. Lund, K. Stølen, and F. Vraalsen, "Model-based security analysis in seven steps - a guided tour to the CORAS method," *BT Technology Journal*, vol. 25, no. 1, pp. 101–117, 2007.

[33] G. Brndeland, H. E. Dahl, I. Engan, and K. Stölen, "Using Dependent CORAS Diagrams to Analyse Mutual Dependency," in *CRITIS'2007*, ser. LNCS 5141. Springer Press, 2008, pp. 135–148.

[34] Walton-on-Thames: Insight Consulting, "CRAMM User Guide," July 2005, risk Analysis and Management Method, Version 5.1.

[35] ISO/IEC-27001/27005, "Information technology. Security techniques. (27001) Information security management systems; (27005) Information security risk management." 2008.

[36] AS/NZS-4360:2004, "Australian/New Zealand Standards, Risk Management," Sydney, NSW, 2004.

[37] P. H. Engebretson and J. J. Pauli, "Leveraging Parent Mitigations and Threats for CAPEC-Driven Hierarchies," in *ITNG'09*. IEEE Press, 2009, pp. 344–349.

[38] Y. Yu, T. T. Tun, A. Tedeschi, V. N. L. Franqueira, and B. Nuseibeh, "Openargue: Supporting argumentation to evolve secure software systems," in *RE'11*. IEEE press, 2011.

[39] "Common Weakness Scoring System (CWSS)," Online: http://cwe.mitre.org/cwss/#vectors, 2011, version 0.2, 14 February 2011.

[40] V. N. L. Franqueira, S. Houmb, and M. Daneva, "Using Real Option Thinking to Improve Decision Making in Security Investment," in *IS'2010 (OTM Conferences)*, ser. LNCS. Springer Press, 2010, pp. 619–638.

[41] R. Haenni, B. Anrig, J. Kohlas, and N. Lehmann, "A Survey on Probabilistic Argumentation," in *ECSQARU'01*, 2001, pp. 19–25.